OpenCMISS-iron examples and tests used by *OpenCMISS* developers at University of Stuttgart, Germany

Christian Bleiler[*], Andreas Hessenthaler[*], Thomas Klotz[*], Aaron Krämer[†], Benjamin Maier[‡], Sergio Morales[*], Mylena Mordhorst[*], Harry Saini[*]

June 30, 2017
17:15

CONTENTS

* Institute of Applied Mechanics (CE), University of Stuttgart, Pfaffenwaldring 7, 70569 Stuttgart, Germany
† Institute for Parallel and Distributed Systems, University of Stuttgart, Universitätsstraße 38, 70569 Stuttgart, Germany
‡ Lehrstuhl Mathematische Methoden für komplexe Simulation der Naturwissenschaft und Technik, University of Stuttgart, Allmandring 5b, 70569 Stuttgart, Germany

LIST OF FIGURES

LIST OF TABLES

1 INTRODUCTION

This document contains information about examples used for testing *OpenCMISS-iron*. Read: How-to[1] and [1].

1.1 Cmgui files for cmgui-2.9

1.2 Variations to consider

- Geometry and topology
  - 1D, 2D, 3D
  - Length, width, height
  - Number of elements
  - Interpolation order
  - Generated or user meshes
  - quad/hex or tri/tet meshes
- Initial conditions
- Load cases
  - Dirichlet BC
  - Neumann BC
  - Volume force
  - Mix of previous items
- Sources, sinks
- Time dependence
  - Static
  - Quasi-static
  - Dynamic
- Material laws
  - Linear
  - Nonlinear (Mooney-Rivlin, Neo-Hookean, Ogden, etc.)
  - Active (Stress, strain)
- Material parameters, anisotropy
- Solver
  - Direct
  - Iterative
- Test cases
  - Numerical reference data
  - Analytical solution
- A mix of previous items

1 https://bitbucket.org/hessenthaler/opencmiss-howto

1.3 Folder structure

TBD..

2 HOW TO WORK ON THIS DOCUMENT

In the Google Doc at https://docs.google.com/spreadsheets/d/1RGKj8vVPqQ-PH0UwMX_e9TAzqaYavKiOzOD4pKY9RGI/edit#gid=0 please indicate what you are working on or if a given example was finished

- no mark: to be done
- x: currently working on it
- xx: done

Initials	Full name
CB	Christian Bleiler
AH	Andreas Hessenthaler
TK	Thomas Klotz
AK	Aaron Krämer
BM	Benjamin Maier
SM	Sergio Morales
MM	Mylena Mordhorst
HS	Harry Saini

Table 1: Initials of people working on examples, in alphabetical order (surnames).

3 DIFFUSION EQUATION

3.1 Equation in general form

The governing equation is,

$$\partial_t u + \nabla \cdot [\boldsymbol{\sigma} \nabla u] = f, \tag{1}$$

with conductivity tensor $\boldsymbol{\sigma}$. The conductivity tensor is,

- defined in material coordinates (fibre direction),
- diagonal,
- defined per element.

3.2 Example-0001

Example uses generated regular meshes and solves a static problem, i.e., applies the boundary conditions in one step.

3.2.1 *Mathematical model - 2D*

We solve the following scalar equation,

$$\nabla \cdot \nabla u = 0 \qquad \Omega = [0,2] \times [0,1], \tag{2}$$

with boundary conditions

$$u = 0 \qquad x = y = 0, \tag{3}$$
$$u = 1 \qquad x = 2, y = 1. \tag{4}$$

No material parameters to specify.

3.2.2 *Mathematical model - 3D*

We solve the following scalar equation,

$$\nabla \cdot \nabla u = 0 \qquad \Omega = [0,2] \times [0,1] \times [0,1], \tag{5}$$

with boundary conditions

$$u = 0 \qquad x = y = z = 0, \tag{6}$$
$$u = 1 \qquad x = 2, y = z = 1. \tag{7}$$

No material parameters to specify.

3.2.3 *Computational model*

- Commandline arguments are:

  float: length along x-direction

  float: length along y-direction

  float: length along z-direction (set to zero for 2D)

  integer: number of elements in x-direction

  integer: number of elements in y-direction

  integer: number of elements in z-direction (set to zero for 2D)

  interger: interpolation order (1: linear; 2: quadratic)

  integer: solver type (0: direct; 1: iterative)

- Commandline arguments for tests are:

  2.0 1.0 0.0 2 1 0 1 0

  2.0 1.0 0.0 4 2 0 1 0

  2.0 1.0 0.0 8 4 0 1 0

  2.0 1.0 0.0 2 1 0 2 0

  2.0 1.0 0.0 4 2 0 2 0

  2.0 1.0 0.0 8 4 0 2 0

  2.0 1.0 0.0 2 1 0 1 1

  2.0 1.0 0.0 4 2 0 1 1

2.0 1.0 0.0 8 4 0 1 1

2.0 1.0 0.0 2 1 0 2 1

2.0 1.0 0.0 4 2 0 2 1

2.0 1.0 0.0 8 4 0 2 1

2.0 1.0 1.0 2 1 1 1 0

2.0 1.0 1.0 4 2 2 1 0

2.0 1.0 1.0 8 4 4 1 0

2.0 1.0 1.0 2 1 1 2 0

2.0 1.0 1.0 4 2 2 2 0

2.0 1.0 1.0 8 4 4 2 0

2.0 1.0 1.0 2 1 1 1 1

2.0 1.0 1.0 4 2 2 1 1

2.0 1.0 1.0 8 4 4 1 1

2.0 1.0 1.0 2 1 1 2 1

2.0 1.0 1.0 4 2 2 2 1

2.0 1.0 1.0 8 4 4 2 1

3.2.4 *Result summary*

We use CHeart rev. 6292 to produce numerical reference solutions.

```
Passed tests: 24 / 24
```

```
No failed tests.
```

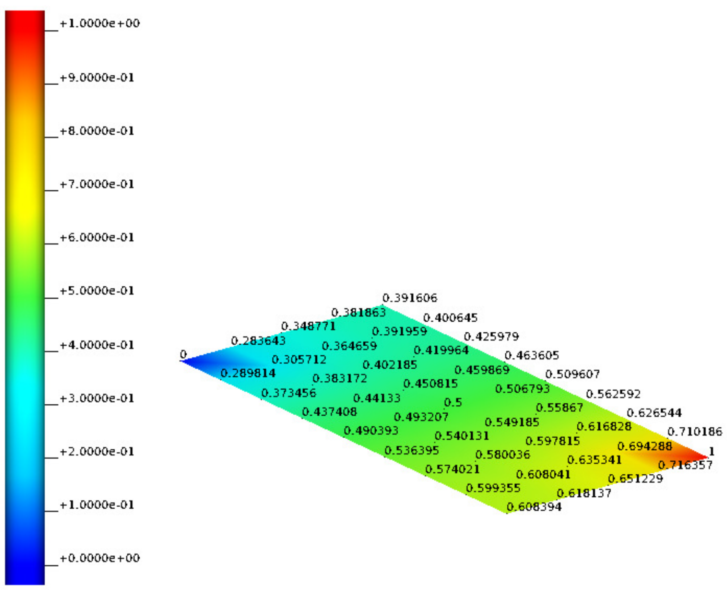

Figure 1: 2D results, iron reference w/ command line arguments [2.0 1.0 0.0 8 4 0 1 0].

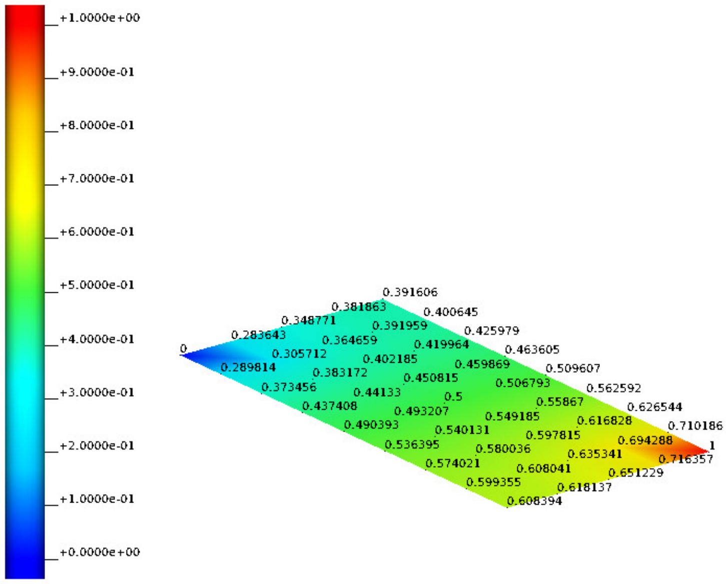

Figure 2: 2D results, current run w/ command line arguments [2.0 1.0 0.0 8 4 0 1 0].

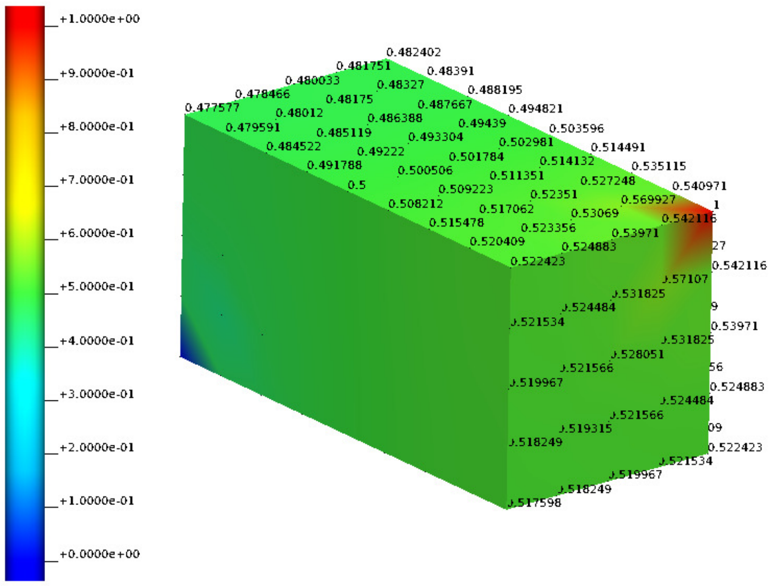

Figure 3: 3D results, iron reference w/ command line arguments [2.0 1.0 1.0 8 4 4 1 0].

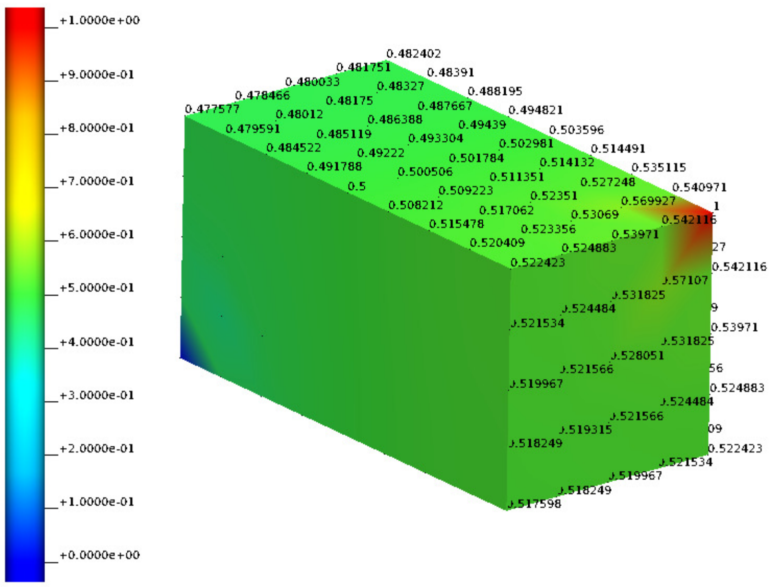

Figure 4: 3D results, current run w/ command line arguments [2.0 1.0 1.0 8 4 4 1 0].

3.3 Example-0001-u

Example uses user-defined regular meshes in CHeart mesh format and solves a static problem, i.e., applies the boundary conditions in one step.

3.3.1 *Mathematical model - 2D*

We solve the following scalar equation,

$$\nabla \cdot \nabla u = 0 \qquad \Omega = [0,2] \times [0,1], \tag{8}$$

with boundary conditions

$$u = 0 \qquad x = y = 0, \tag{9}$$
$$u = 1 \qquad x = 2, y = 1. \tag{10}$$

No material parameters to specify.

3.3.2 *Mathematical model - 3D*

We solve the following scalar equation,

$$\nabla \cdot \nabla u = 0 \qquad \Omega = [0,2] \times [0,1] \times [0,1], \tag{11}$$

with boundary conditions

$$u = 0 \qquad x = y = z = 0, \tag{12}$$
$$u = 1 \qquad x = 2, y = z = 1. \tag{13}$$

No material parameters to specify.

3.3.3 *Computational model*

- Commandline arguments are:
  - float: length along x-direction
  - float: length along y-direction
  - float: length along z-direction (set to zero for 2D)
  - integer: number of elements in x-direction
  - integer: number of elements in y-direction
  - integer: number of elements in z-direction (set to zero for 2D)
  - interger: interpolation order (1: linear; 2: quadratic)
  - integer: solver type (0: direct; 1: iterative)
- Commandline arguments for tests are:
  - 2.0 1.0 0.0 2 1 0 1 0
  - 2.0 1.0 0.0 4 2 0 1 0
  - 2.0 1.0 0.0 8 4 0 1 0
  - 2.0 1.0 0.0 2 1 0 2 0
  - 2.0 1.0 0.0 4 2 0 2 0
  - 2.0 1.0 0.0 8 4 0 2 0
  - 2.0 1.0 0.0 2 1 0 1 1
  - 2.0 1.0 0.0 4 2 0 1 1

2.0 1.0 0.0 8 4 0 1 1

2.0 1.0 0.0 2 1 0 2 1

2.0 1.0 0.0 4 2 0 2 1

2.0 1.0 0.0 8 4 0 2 1

2.0 1.0 1.0 2 1 1 1 0

2.0 1.0 1.0 4 2 2 1 0

2.0 1.0 1.0 8 4 4 1 0

2.0 1.0 1.0 2 1 1 2 0

2.0 1.0 1.0 4 2 2 2 0

2.0 1.0 1.0 8 4 4 2 0

2.0 1.0 1.0 2 1 1 1 1

2.0 1.0 1.0 4 2 2 1 1

2.0 1.0 1.0 8 4 4 1 1

2.0 1.0 1.0 2 1 1 2 1

2.0 1.0 1.0 4 2 2 2 1

2.0 1.0 1.0 8 4 4 2 1

- Note: Binary uses command line arguments to search for the relevant mesh files.

3.3.4 *Result summary*

We use CHeart rev. 6292 to produce numerical reference solutions.

```
Passed tests: 24 / 24

No failed tests.
```

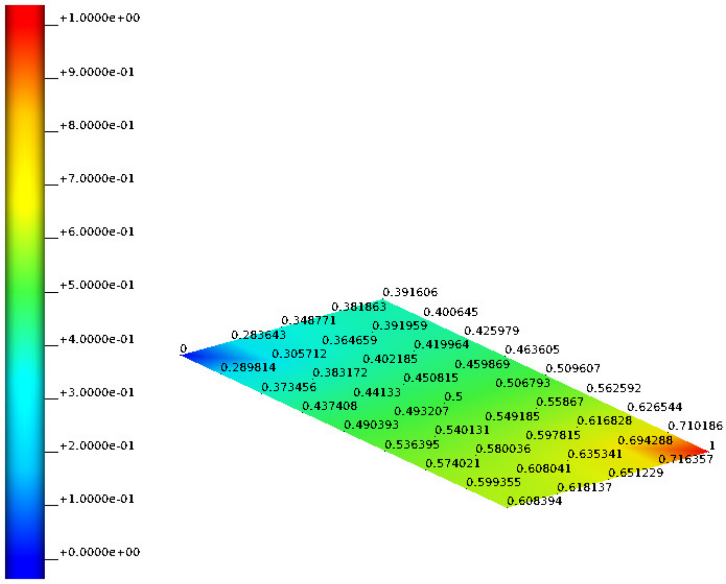

Figure 5: 2D results, iron reference w/ command line arguments [2.0 1.0 0.0 8 4 0 1 0].

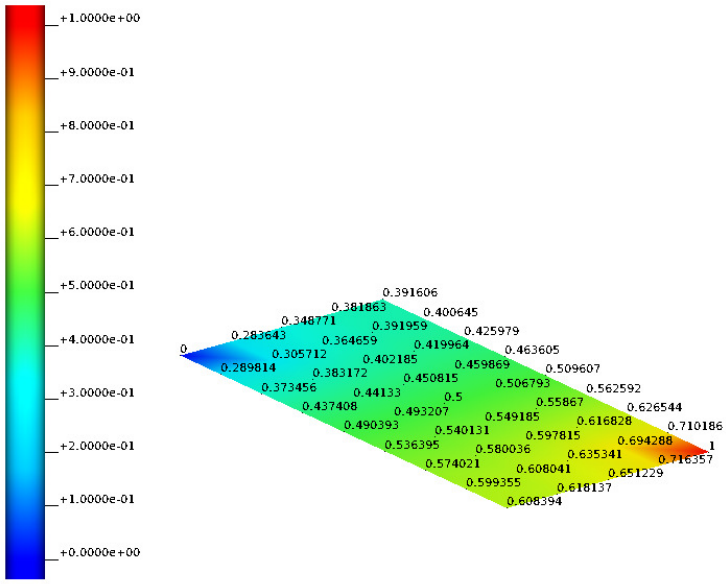

Figure 6: 2D results, current run w/ command line arguments [2.0 1.0 0.0 8 4 0 1 0].

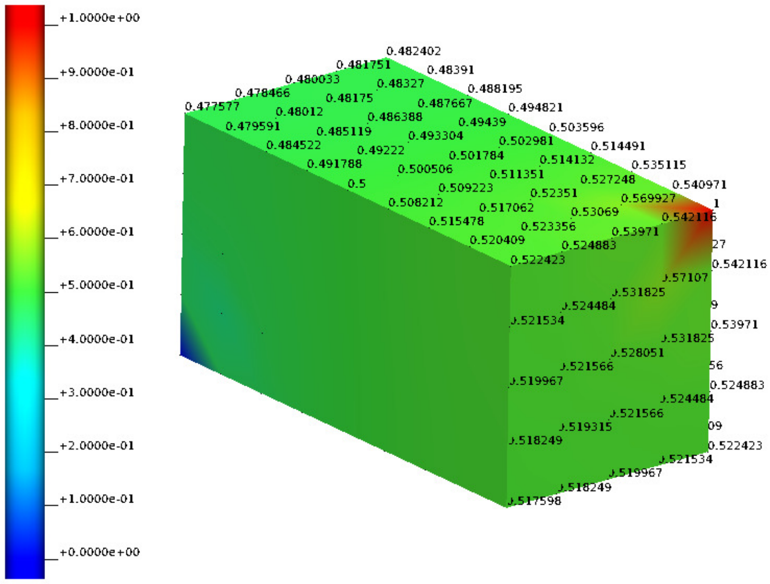

Figure 7: 3D results, iron reference w/ command line arguments [2.0 1.0 1.0 8 4 4 1 0].

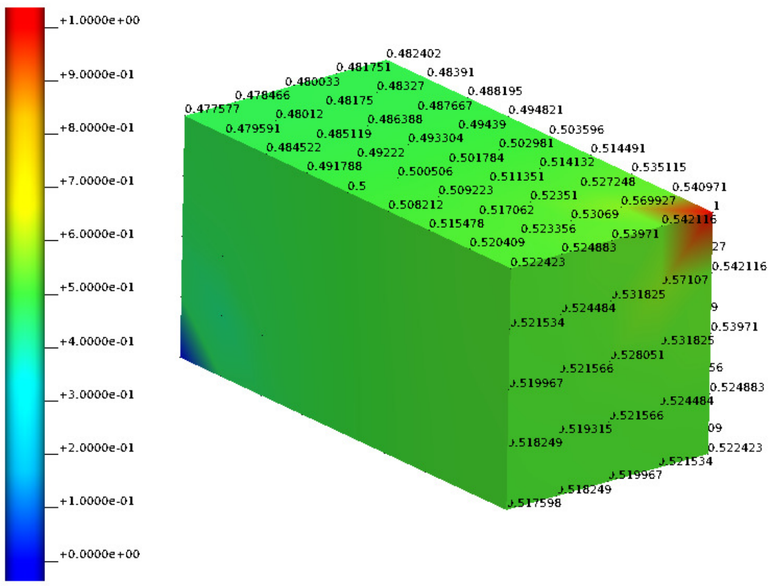

Figure 8: 3D results, current run w/ command line arguments [2.0 1.0 1.0 8 4 4 1 0].

3.4 Example-0002

Example uses generated regular meshes and solves a static problem, i.e., applies the boundary conditions in one step.

3.4.1 *Mathematical model - 2D*

We solve the following scalar equation,

$$\nabla \cdot \nabla u = 0 \qquad \Omega = [0,2] \times [0,1], \tag{14}$$

with boundary conditions

$$u = 15y \qquad x = 0, \tag{15}$$

$$u = 25 - 18y \qquad x = 2. \tag{16}$$

No material parameters to specify.

3.4.2 *Mathematical model - 3D*

We solve the following scalar equation,

$$\nabla \cdot \nabla u = 0 \qquad \Omega = [0,2] \times [0,1] \times [0,1], \tag{17}$$

with boundary conditions

$$u = 15y \qquad x = 0, \tag{18}$$

$$u = 25 - 18y \qquad x = 2. \tag{19}$$

No material parameters to specify.

3.4.3 *Computational model*

- Commandline arguments are:

  float: length along x-direction
  float: length along y-direction
  float: length along z-direction (set to zero for 2D)
  integer: number of elements in x-direction
  integer: number of elements in y-direction
  integer: number of elements in z-direction (set to zero for 2D)
  interger: interpolation order (1: linear; 2: quadratic)
  integer: solver type (0: direct; 1: iterative)

- Commandline arguments for tests are:

  2.0 1.0 0.0 2 1 0 1 0
  2.0 1.0 0.0 4 2 0 1 0
  2.0 1.0 0.0 8 4 0 1 0
  2.0 1.0 0.0 2 1 0 2 0
  2.0 1.0 0.0 4 2 0 2 0
  2.0 1.0 0.0 8 4 0 2 0
  2.0 1.0 0.0 2 1 0 1 1
  2.0 1.0 0.0 4 2 0 1 1

2.0 1.0 0.0 8 4 0 1 1

2.0 1.0 0.0 2 1 0 2 1

2.0 1.0 0.0 4 2 0 2 1

2.0 1.0 0.0 8 4 0 2 1

2.0 1.0 1.0 2 1 1 1 0

2.0 1.0 1.0 4 2 2 1 0

2.0 1.0 1.0 8 4 4 1 0

2.0 1.0 1.0 2 1 1 2 0

2.0 1.0 1.0 4 2 2 2 0

2.0 1.0 1.0 8 4 4 2 0

2.0 1.0 1.0 2 1 1 1 1

2.0 1.0 1.0 4 2 2 1 1

2.0 1.0 1.0 8 4 4 1 1

2.0 1.0 1.0 2 1 1 2 1

2.0 1.0 1.0 4 2 2 2 1

2.0 1.0 1.0 8 4 4 2 1

3.4.4 *Result summary*

We use CHeart rev. 6292 to produce numerical reference solutions.

```
Passed tests: 24 / 24
```

```
No failed tests.
```

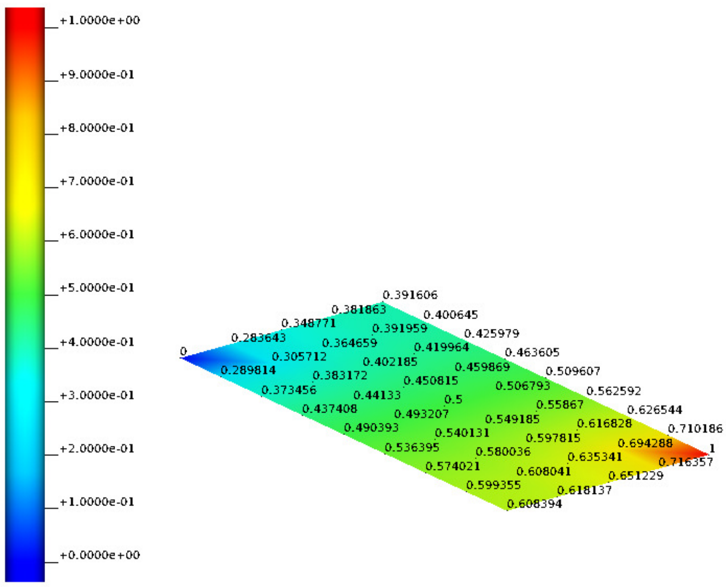

Figure 9: 2D results, iron reference w/ command line arguments [2.0 1.0 0.0 8 4 0 1 0].

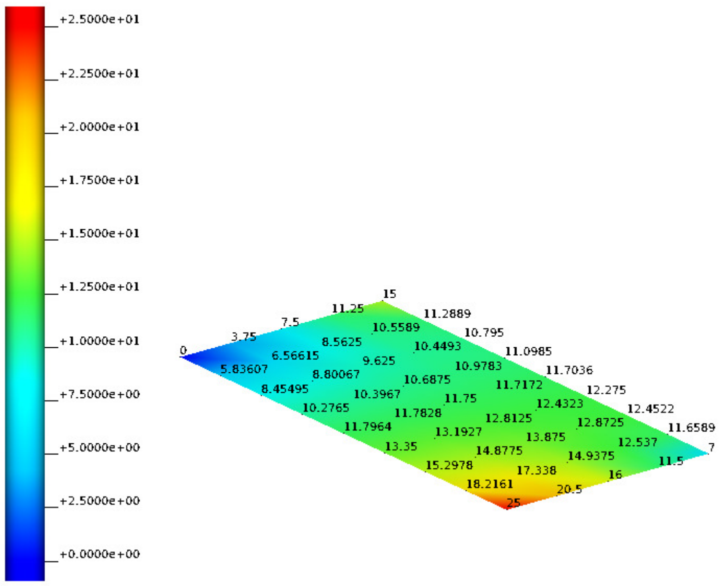

Figure 10: 2D results, current run w/ command line arguments [2.0 1.0 0.0 8 4 0 1 0].

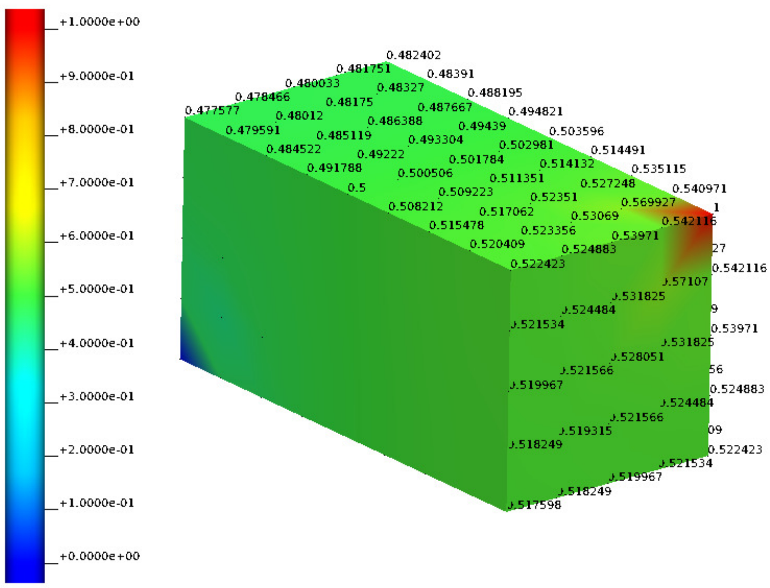

Figure 11: 3D results, iron reference w/ command line arguments [2.0 1.0 1.0 8 4 4 1 0].

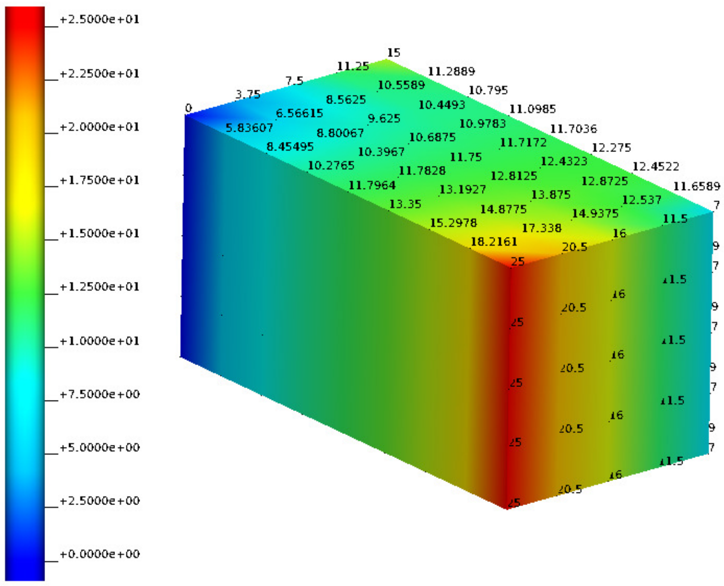

Figure 12: 3D results, current run w/ command line arguments [2.0 1.0 1.0 8 4 4 1 0].

3.5 Example-0003

Example uses generated regular meshes and solves a static problem, i.e., applies the boundary conditions in one step.

3.5.1 *Mathematical model - 2D*

We solve the following scalar equation,

$$\nabla \cdot \nabla u = 0 \qquad \Omega = [0,2] \times [0,1], \tag{20}$$

with boundary conditions

$$u = 15y \qquad x = 0, \tag{21}$$
$$\partial_n u = 25 - 18y \qquad x = 2. \tag{22}$$

No material parameters to specify.

3.5.2 *Mathematical model - 3D*

We solve the following scalar equation,

$$\nabla \cdot \nabla u = 0 \qquad \Omega = [0,2] \times [0,1] \times [0,1], \tag{23}$$

with boundary conditions

$$u = 15y \qquad x = 0, \tag{24}$$
$$\partial_n u = 25 - 18y \qquad x = 2. \tag{25}$$

No material parameters to specify.

3.5.3 *Computational model*

- Commandline arguments are:

  float: length along x-direction
  float: length along y-direction
  float: length along z-direction (set to zero for 2D)
  integer: number of elements in x-direction
  integer: number of elements in y-direction
  integer: number of elements in z-direction (set to zero for 2D)
  interger: interpolation order (1: linear; 2: quadratic)
  integer: solver type (0: direct; 1: iterative)

- Commandline arguments for tests are:

  2.0 1.0 0.0 2 1 0 1 0
  2.0 1.0 0.0 4 2 0 1 0
  2.0 1.0 0.0 8 4 0 1 0
  2.0 1.0 0.0 2 1 0 2 0
  2.0 1.0 0.0 4 2 0 2 0
  2.0 1.0 0.0 8 4 0 2 0
  2.0 1.0 0.0 2 1 0 1 1
  2.0 1.0 0.0 4 2 0 1 1

2.0 1.0 0.0 8 4 0 1 1

2.0 1.0 0.0 2 1 0 2 1

2.0 1.0 0.0 4 2 0 2 1

2.0 1.0 0.0 8 4 0 2 1

2.0 1.0 1.0 2 1 1 1 0

2.0 1.0 1.0 4 2 2 1 0

2.0 1.0 1.0 8 4 4 1 0

2.0 1.0 1.0 2 1 1 2 0

2.0 1.0 1.0 4 2 2 2 0

2.0 1.0 1.0 8 4 4 2 0

2.0 1.0 1.0 2 1 1 1 1

2.0 1.0 1.0 4 2 2 1 1

2.0 1.0 1.0 8 4 4 1 1

2.0 1.0 1.0 2 1 1 2 1

2.0 1.0 1.0 4 2 2 2 1

2.0 1.0 1.0 8 4 4 2 1

3.5.4 *Result summary*

We use CHeart rev. 6292 to produce numerical reference solutions.

```
Passed tests: 0 / 24

Failed tests:
current_run/l2x1x0_n2x1x0_i1_s0/Example.part0.exnode      | CHeart   - Iron |_2 = 44.2627
current_run/l2x1x0_n4x2x0_i1_s0/Example.part0.exnode      | CHeart   - Iron |_2 = 37.2770
current_run/l2x1x0_n8x4x0_i1_s0/Example.part0.exnode      | CHeart   - Iron |_2 = 32.2165
current_run/l2x1x0_n2x1x0_i2_s0/Example.part0.exnode      | CHeart   - Iron |_2 = 27.3358
current_run/l2x1x0_n4x2x0_i2_s0/Example.part0.exnode      | CHeart   - Iron |_2 = 22.1869
current_run/l2x1x0_n8x4x0_i2_s0/Example.part0.exnode      | CHeart   - Iron |_2 = 19.7449
current_run/l2x1x0_n2x1x0_i1_s1/Example.part0.exnode      | CHeart   - Iron |_2 = 44.2627
current_run/l2x1x0_n4x2x0_i1_s1/Example.part0.exnode      | CHeart   - Iron |_2 = 37.2770
current_run/l2x1x0_n8x4x0_i1_s1/Example.part0.exnode      | CHeart   - Iron |_2 = 32.2165
current_run/l2x1x0_n2x1x0_i2_s1/Example.part0.exnode      | CHeart   - Iron |_2 = 27.3358
current_run/l2x1x0_n4x2x0_i2_s1/Example.part0.exnode      | CHeart   - Iron |_2 = 22.1869
current_run/l2x1x0_n8x4x0_i2_s1/Example.part0.exnode      | CHeart   - Iron |_2 = 19.7449
current_run/l2x1x1_n2x1x1_i1_s0/Example.part0.exnode      | CHeart   - Iron |_2 = 124.749
current_run/l2x1x1_n4x2x2_i1_s0/Example.part0.exnode      | CHeart   - Iron |_2 = 128.672
current_run/l2x1x1_n8x4x4_i1_s0/Example.part0.exnode      | CHeart   - Iron |_2 = 143.606
current_run/l2x1x1_n2x1x1_i2_s0/Example.part0.exnode      | CHeart   - Iron |_2 = 94.2619
current_run/l2x1x1_n4x2x2_i2_s0/Example.part0.exnode      | CHeart   - Iron |_2 = 98.7606
current_run/l2x1x1_n8x4x4_i2_s0/Example.part0.exnode      | CHeart   - Iron |_2 = 118.047
current_run/l2x1x1_n2x1x1_i1_s1/Example.part0.exnode      | CHeart   - Iron |_2 = 124.749
current_run/l2x1x1_n4x2x2_i1_s1/Example.part0.exnode      | CHeart   - Iron |_2 = 128.672
current_run/l2x1x1_n8x4x4_i1_s1/Example.part0.exnode      | CHeart   - Iron |_2 = 143.606
current_run/l2x1x1_n2x1x1_i2_s1/Example.part0.exnode      | CHeart   - Iron |_2 = 94.2619
current_run/l2x1x1_n4x2x2_i2_s1/Example.part0.exnode      | CHeart   - Iron |_2 = 98.7606
current_run/l2x1x1_n8x4x4_i2_s1/Example.part0.exnode      | CHeart   - Iron |_2 = 118.047
```

Figure 13: 2D results, iron reference w/ command line arguments [2.0 1.0 0.0 8 4 0 1 0].

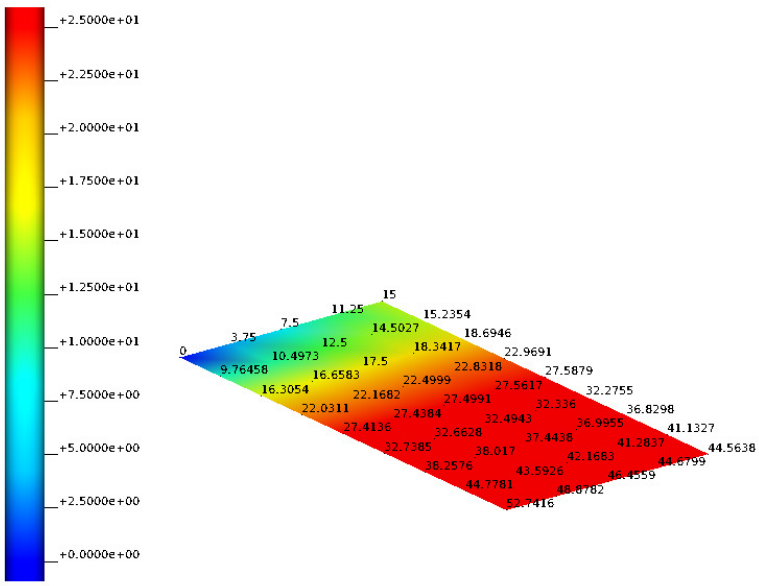

Figure 14: 2D results, current run w/ command line arguments [2.0 1.0 0.0 8 4 0 1 0].

Figure 15: 3D results, iron reference w/ command line arguments [2.0 1.0 1.0 8 4 4 1 0].

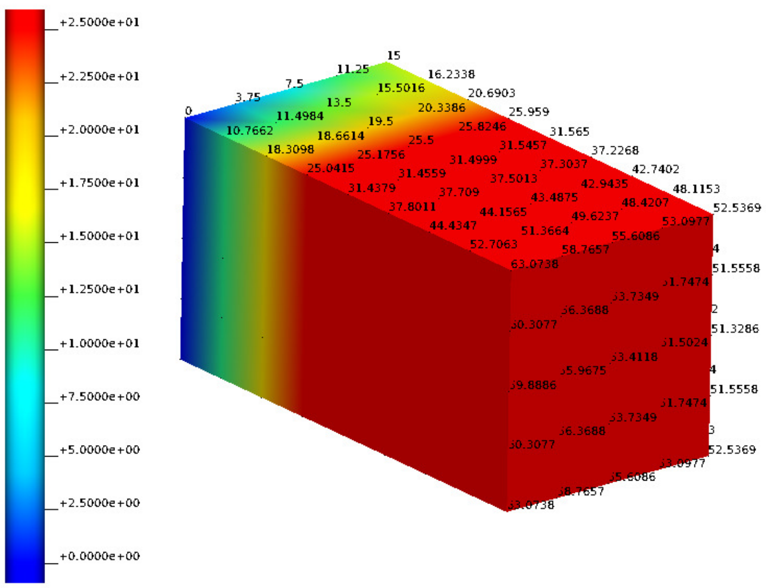

Figure 16: 3D results, current run w/ command line arguments [2.0 1.0 1.0 8 4 4 1 0].

3.6 Example-0004

Example uses generated regular meshes and solves a static problem, i.e., applies the boundary conditions in one step.

3.6.1 *Mathematical model - 2D*

We solve the following scalar equation,

$$\nabla \cdot \nabla u = 0 \qquad\qquad \Omega = [0,2] \times [0,1], \tag{26}$$

with boundary conditions

$$u = 2.0e^{x} \cdot \cos(y) \qquad\qquad \text{on } \partial\Omega. \tag{27}$$

No material parameters to specify.

3.6.2 *Computational model*

- Commandline arguments are:
  integer: number of elements in x-direction
  integer: number of elements in y-direction
  integer: number of elements in z-direction (set to zero for 2D)
  interger: interpolation order (1: linear; 2: quadratic)
  integer: solver type (0: direct; 1: iterative)
- Commandline arguments for tests are:
  4 2 0 1 0
  8 4 0 1 0
  2 1 0 2 0
  4 2 0 2 0
  8 4 0 2 0
  4 2 0 1 1
  8 4 0 1 1
  2 1 0 2 1
  4 2 0 2 1
  8 4 0 2 1
  100 50 0 1 0 (not tested yet..)
  100 50 0 2 0 (not tested yet..)
  100 50 0 1 1 (not tested yet..)
  100 50 0 2 1 (not tested yet..)

3.6.3 *Result summary*

We use CHeart rev. 6292 to produce numerical reference solutions.

```
Passed tests: 10 / 10
```

```
No failed tests.
```

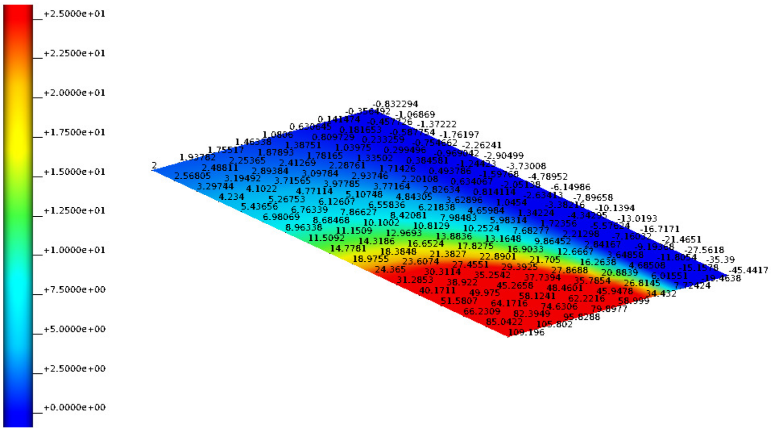

Figure 17: 2D results, iron reference w/ command line arguments [8 4 0 2 0].

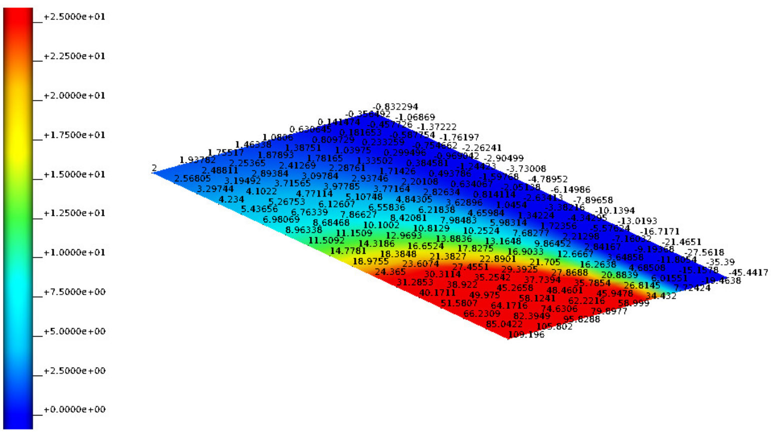

Figure 18: 2D results, current run w/ command line arguments [8 4 0 2 0].

3.7 Example-0011

Example uses generated regular meshes and solves a static problem, i.e., applies the boundary conditions in one step.

3.7.1 *Mathematical model - 2D*

We solve the following scalar equation,

$$\nabla \cdot [\boldsymbol{\sigma} \nabla u] = 0 \qquad \Omega = [0, 2] \times [0, 1], \tag{28}$$

with boundary conditions

$$u = 0 \qquad x = y = 0, \tag{29}$$

$$u = 1 \qquad x = 2, y = 1. \tag{30}$$

The conductivity tensor is defined as,

$$\boldsymbol{\sigma}(x, t) = \boldsymbol{\sigma} = \mathbf{I}. \tag{31}$$

3.7.2 *Mathematical model - 3D*

We solve the following scalar equation,

$$\nabla \cdot [\boldsymbol{\sigma} \nabla u] = 0 \qquad \Omega = [0, 2] \times [0, 1] \times [0, 1], \tag{32}$$

with boundary conditions

$$u = 0 \qquad x = y = z = 0, \tag{33}$$

$$u = 1 \qquad x = 2, y = z = 1. \tag{34}$$

The conductivity tensor is defined as,

$$\boldsymbol{\sigma}(x, t) = \boldsymbol{\sigma} = \mathbf{I}. \tag{35}$$

3.7.3 *Computational model*

- Commandline arguments are:

  float: length along x-direction

  float: length along y-direction

  float: length along z-direction (set to zero for 2D)

  integer: number of elements in x-direction

  integer: number of elements in y-direction

  integer: number of elements in z-direction (set to zero for 2D)

  integer: interpolation order (1: linear; 2: quadratic)

  integer: solver type (0: direct; 1: iterative)

  float: σ_{11}

  float: σ_{22}

  float: σ_{33} (ignored for 2D)

- Commandline arguments for tests are:

  2.0 1.0 0.0 2 1 0 1 0 1 1

  2.0 1.0 0.0 4 2 0 1 0 1 1

  2.0 1.0 0.0 8 4 0 1 0 1 1

  2.0 1.0 0.0 2 1 0 2 0 1 1

  2.0 1.0 0.0 4 2 0 2 0 1 1

  2.0 1.0 0.0 8 4 0 2 0 1 1

  2.0 1.0 0.0 2 1 0 1 1 1 1

  2.0 1.0 0.0 4 2 0 1 1 1 1

  2.0 1.0 0.0 8 4 0 1 1 1 1

  2.0 1.0 0.0 2 1 0 2 1 1 1

  2.0 1.0 0.0 4 2 0 2 1 1 1

  2.0 1.0 0.0 8 4 0 2 1 1 1

  2.0 1.0 1.0 2 1 1 1 0 1 1 1

  2.0 1.0 1.0 4 2 2 1 0 1 1 1

  2.0 1.0 1.0 8 4 4 1 0 1 1 1

  2.0 1.0 1.0 2 1 1 2 0 1 1 1

  2.0 1.0 1.0 4 2 2 2 0 1 1 1

  2.0 1.0 1.0 8 4 4 2 0 1 1 1

  2.0 1.0 1.0 2 1 1 1 1 1 1 1

  2.0 1.0 1.0 4 2 2 1 1 1 1 1

  2.0 1.0 1.0 8 4 4 1 1 1 1 1

  2.0 1.0 1.0 2 1 1 2 1 1 1 1

  2.0 1.0 1.0 4 2 2 2 1 1 1 1

  2.0 1.0 1.0 8 4 4 2 1 1 1 1

3.7.4 *Result summary*

We use CHeart rev. 6292 to produce numerical reference solutions.

```
Passed tests: 24 / 24
```

```
No failed tests.
```

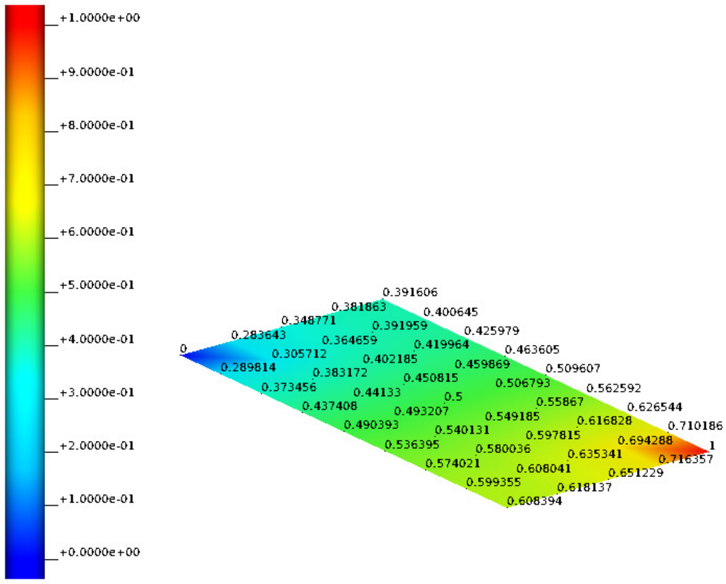

Figure 19: 2D results, iron reference w/ command line arguments [2.0 1.0 0.0 8 4 0 1 0 1 1].

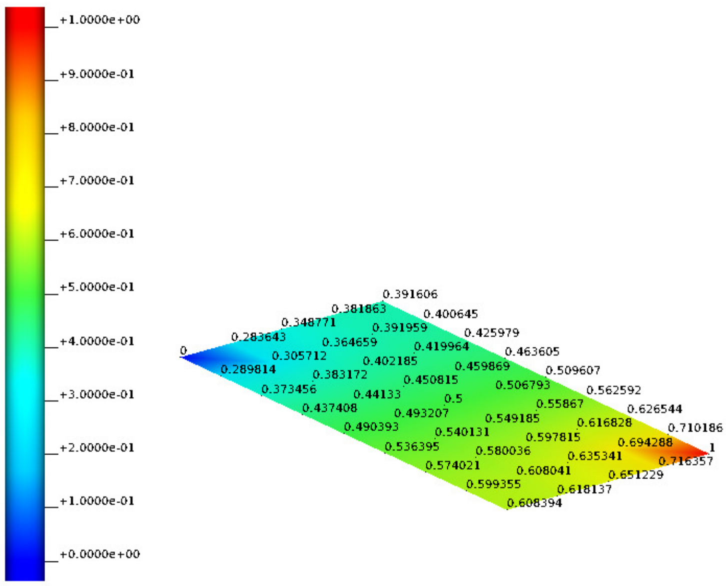

Figure 20: 2D results, current run w/ command line arguments [2.0 1.0 0.0 8 4 0 1 0 1 1].

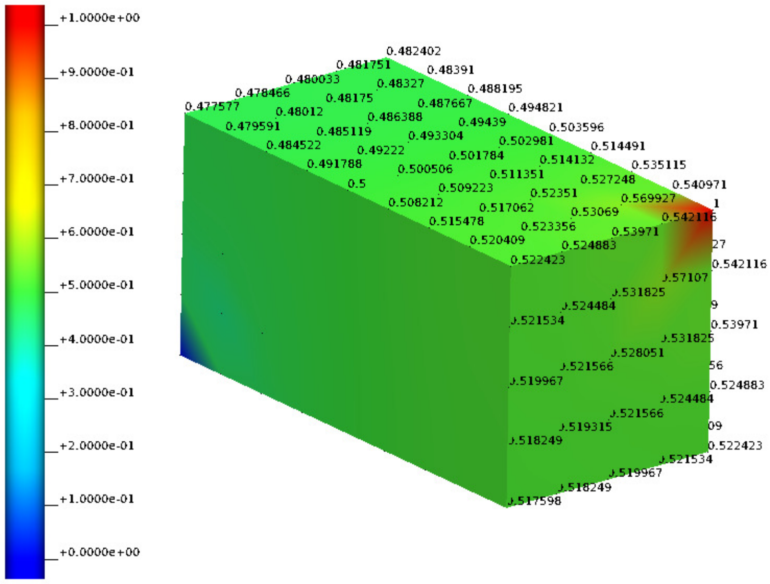

Figure 21: 3D results, iron reference w/ command line arguments [2.0 1.0 1.0 8 4 4 1 0 1 1 1].

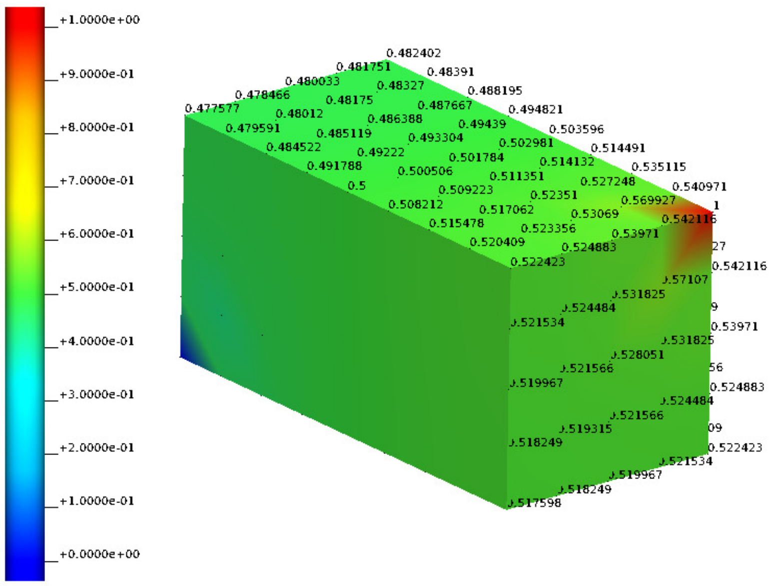

Figure 22: 3D results, current run w/ command line arguments [2.0 1.0 1.0 8 4 4 1 0 1 1 1].

4 LINEAR ELASTICITY

4.1 Equation in general form

$$\partial_{tt}\mathbf{u} + \nabla \cdot \boldsymbol{\sigma}(\mathbf{u}, t) = \mathbf{f}(\mathbf{u}, t) \tag{36}$$

4.2 Example-0101

4.2.1 *Mathematical model*

We solve the following equation,

$$\nabla \cdot \boldsymbol{\sigma}(\mathbf{u}, t) = 0 \qquad \Omega = [0, 160] \times [0, 120], t \in [0, 5], \tag{37}$$

with time step size $\Delta_t = 1$ and boundary conditions

$$\dots \qquad \dots, \tag{38}$$

$$\dots \qquad \dots. \tag{39}$$

2D: specify thickness, Young's modulus and Poisson's ratio.

4.2.2 *Computational model*

- Length, width, height
- Direct/iterative solver
- Generated/user mesh
- Number of elements
- Interpolation order
- Number of solver steps (time steps, load steps)

4.2.3 *Results*

Figure 23: Results, analytical solution.

4.2.4 *Validation*

CHeart rev. 6328, Abaqus 2017, analytical reference solution, whatever...

Figure 24: Results, Abaqus reference.

Figure 25: Results, iron reference.

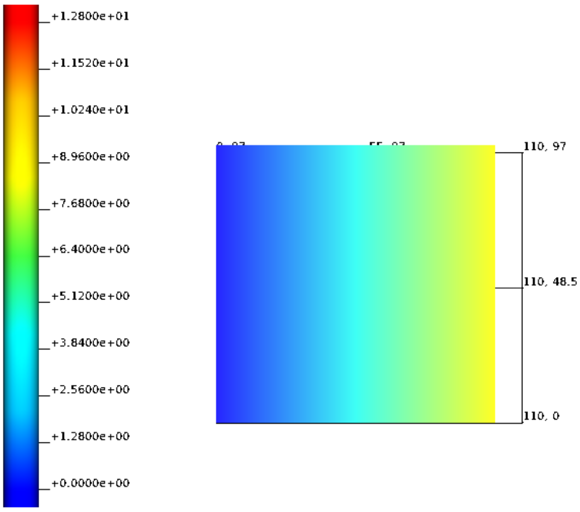

Figure 26: Results, current run.

5 FINITE ELASTICITY

6 NAVIER–STOKES FLOW

7 MONODOMAIN

8 CELLML MODEL

REFERENCES

[1] Chris Bradley, Andy Bowery, Randall Britten, Vincent Budelmann, Oscar Camara, Richard Christie, Andrew Cookson, Alejandro F Frangi, Thiranja Babarenda Gamage, Thomas Heidlauf, et al. Opencmiss: a multi-physics & multi-scale computational infrastructure for the vph/-physiome project. *Progress in biophysics and molecular biology*, 107(1):32–47, 2011.